Non-standard formulations of 3D elasticity with FEniCSx

Andreas Zilian

Department of Engineering | University of Luxembourg

13 June 2024 | FEniCS'24 conference

UNIVERSITÉ DU LUXEMBOURG

Contents

Linear elasticity: displacement/stress (Reissner, TDNNS)

Linear elasticity: stress-only (Beltrami-Michell)

Non-linear elasticity: displacement (spectral representation)

Formulation and discrete function spaces

- work by Pechstein/Schöberl, e.g. [1]
- Hellinger-Reissner, mixed displacement/stress, optional hybridisation
- $\mathbf{u} \in H(\mathrm{curl})$: vector-valued, tangential-continuous
- $\boldsymbol{\sigma} \in H(\mathrm{div}\,\mathrm{div})$: symmetric tensor-valued, normal-normal continuous

Formulation and discrete function spaces

- work by Pechstein/Schöberl, e.g. [1]
- Hellinger-Reissner, mixed displacement/stress, optional hybridisation
- $\mathbf{u} \in H(\mathrm{curl})$: vector-valued, tangential-continuous
- $\boldsymbol{\sigma} \in H(\mathrm{div}\,\mathrm{div})$: symmetric tensor-valued, normal-normal continuous

$$
\begin{aligned}
0 = &\int_\Omega \delta\boldsymbol{\sigma} : (\boldsymbol{\varepsilon}(\boldsymbol{\sigma}) - \boldsymbol{\varepsilon}(\mathbf{u}))\,\mathrm{d}x + \int_{\partial\Omega} \delta\sigma_{nn} u_n\,\mathrm{d}s + \int_{\partial T} \delta\sigma_{nn}[u_n]\,\mathrm{d}S + \int_{\partial\Omega_u^N} \delta\sigma_{nn}\bar{u}_n\,\mathrm{d}s \\
&- \int_\Omega \boldsymbol{\varepsilon}(\delta\mathbf{u}) : \boldsymbol{\sigma}\,\mathrm{d}x + \int_\Omega \delta\mathbf{u}\cdot\mathbf{f}\,\mathrm{d}x + \int_{\partial\Omega} \delta u_n\sigma_{nn}\,\mathrm{d}s + \int_{\partial T} [\delta u_n]\sigma_{nn}\,\mathrm{d}S + \int_{\partial\Omega_t^N} \delta u_t \bar{t}_t\,\mathrm{d}s
\end{aligned}
$$

UNIVERSITÉ DU LUXEMBOURG

Formulation and discrete function spaces

- work by Pechstein/Schöberl, e.g. [1]
- Hellinger-Reissner, mixed displacement/stress, optional hybridisation
- $\mathbf{u} \in H(\mathrm{curl})$: vector-valued, tangential-continuous
- $\boldsymbol{\sigma} \in H(\mathrm{div}\,\mathrm{div})$: symmetric tensor-valued, normal-normal continuous

$$
\begin{aligned}
0 = &\int_\Omega \delta\boldsymbol{\sigma} : (\boldsymbol{\varepsilon}(\boldsymbol{\sigma}) - \boldsymbol{\varepsilon}(\mathbf{u}))\,\mathrm{d}x + \int_{\partial\Omega} \delta\sigma_{nn} u_n\,\mathrm{d}s + \int_{\partial T} \delta\sigma_{nn}[u_n]\,\mathrm{d}S + \int_{\partial\Omega_u^N} \delta\sigma_{nn}\bar{u}_n\,\mathrm{d}s \\
&- \int_\Omega \boldsymbol{\varepsilon}(\delta\mathbf{u}) : \boldsymbol{\sigma}\,\mathrm{d}x + \int_\Omega \delta\mathbf{u}\cdot\mathbf{f}\,\mathrm{d}x + \int_{\partial\Omega} \delta u_n\sigma_{nn}\,\mathrm{d}s + \int_{\partial T} [\delta u_n]\sigma_{nn}\,\mathrm{d}S + \int_{\partial\Omega_t^N} \delta u_t \bar{t}_t\,\mathrm{d}s
\end{aligned}
$$

Challenges

- $H(\mathrm{div}\,\mathrm{div})$ for tetrahedra (PS element)
- Dirichlet boundary conditions: $u_t = \bar{u}_t$ on $\partial\Omega_u^D$ and $\sigma_{nn} = \bar{t}_n$ on $\partial\Omega_t^D$

Pechstein-Schöberl element

- realised as custom Basix element
- triangle, tetrahedron
- arbitrary order $p >= 0$
- straightforward use

```
import basix
import hdivdiv  # custom H(divdiv): HHJ and PS

cell_type = mesh.basix_cell()

Ue = basix.ufl.element("N2E", cell_type, 2)
Se = hdivdiv.create_custom_hdivdiv(cell_type, 2)
```

- desirable:
  more complete access to Basix C++ API

TDNNS | solved challenges

Pechstein-Schöberl element

- realised as custom Basix element
- triangle, tetrahedron
- arbitrary order $p >= 0$
- straightforward use

```python
import basix
import hdivdiv   # custom H(divdiv): HHJ and PS

cell_type = mesh.basix_cell()

Ue = basix.ufl.element("N2E", cell_type, 2)
Se = hdivdiv.create_custom_hdivdiv(cell_type, 2)
```

- desirable:
  more complete access to Basix C++ API

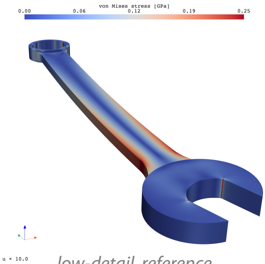
Codimension projection for BCs

- user-defined projector $\mathbf{P}$
- $0 = \int \mathbf{P}(\delta\boldsymbol{\tau}) : (\mathbf{P}(\boldsymbol{\tau}) - \mathcal{E})\,\mathrm{d}s$
- optional regularisation
  $+\varepsilon \int \delta\boldsymbol{\tau} : \boldsymbol{\tau}\,\mathrm{d}s$

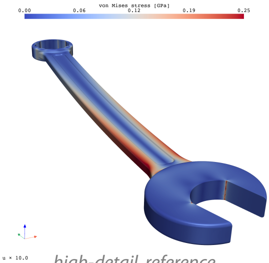
```python
import ufl
import dolfinx
from dolfiny.projection import project_codimension

n = ufl.FacetNormal(mesh)

def normalnormal(A):
        return ufl.dot(ufl.dot(A, n), n)

boundary_mts = ...   # merged MeshTags
boundary_tag = ...   # mesh tag

Sf = dolfinx.fem.functionspace(mesh, Se)
Sn = dolfinx.fem.Function(Sf, name="Sn")

t = ...   # given boundary stress vector
expr = ufl.dot(t, n)   # boundary normal stress

project_codimension(expr, Sn, normalnormal,
                    boundary_mts, boundary_tag)
```

TDNNS | demo

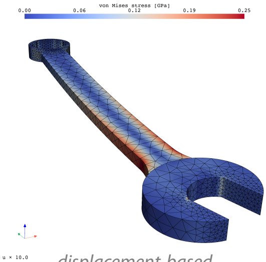
- combination spanner
- fixed flats
- lat. traction at crown



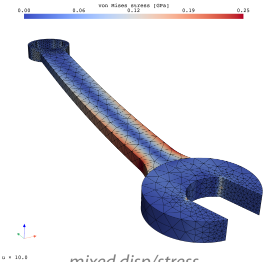
low-detail, reference



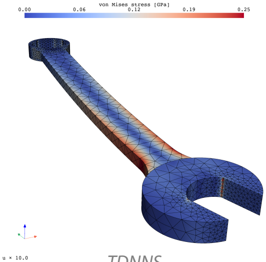
high-detail, reference



displacement-based



mixed disp/stress



TDNNS

Stress-only form | linear elasticity

Formulation and discrete function spaces

- recent work [2] on isotropic case
- symmetrised weak form of Beltrami-Michell equation, stresses only
- derived from equilibrium $\mathbf{0} = \mathrm{div}(\boldsymbol{\sigma}) + \mathbf{f}$ and compatibility $\mathbf{0} = \mathrm{inc}(\boldsymbol{\varepsilon}(\boldsymbol{\sigma}))$
- $\boldsymbol{\sigma} \in H^1$: symmetric tensor-valued

Stress-only form | linear elasticity

Formulation and discrete function spaces

- recent work [2] on isotropic case
- symmetrised weak form of Beltrami-Michell equation, stresses only
- derived from equilibrium $\mathbf{0} = \mathrm{div}(\boldsymbol{\sigma}) + \mathbf{f}$ and compatibility $\mathbf{0} = \mathrm{inc}(\boldsymbol{\varepsilon}(\boldsymbol{\sigma}))$
- $\boldsymbol{\sigma} \in H^1$: symmetric tensor-valued

$$
\begin{aligned}
0 = & \frac{1+\nu}{E} \int_\Omega \mathrm{grad}(\delta\boldsymbol{\sigma}) : \mathrm{grad}(\boldsymbol{\sigma}) - \delta\boldsymbol{\sigma} : 2\mathrm{sym}(\mathrm{grad}(\mathbf{f}))\, \mathrm{d}x \\
& + \frac{1}{E} \int_\Omega \mathrm{div}(\delta\boldsymbol{\sigma}) : \mathrm{grad}(\mathrm{tr}(\boldsymbol{\sigma})) + \mathrm{grad}(\mathrm{tr}(\delta\boldsymbol{\sigma})) : \mathrm{div}(\boldsymbol{\sigma})\, \mathrm{d}x \\
& - \frac{1+\nu^2}{E(1-\nu)} \int_\Omega \mathrm{tr}(\delta\boldsymbol{\sigma})\, \mathrm{div}(\mathbf{f})\, \mathrm{d}x - \frac{1}{E} \int_{\partial\Omega} \delta\boldsymbol{\sigma} : \bar{\boldsymbol{\tau}}\, \mathrm{d}s + \text{stab}
\end{aligned}
$$

Stress-only form | linear elasticity

Formulation and discrete function spaces

- recent work [2] on isotropic case
- symmetrised weak form of Beltrami-Michell equation, stresses only
- derived from equilibrium $\mathbf{0} = \mathrm{div}(\boldsymbol{\sigma}) + \mathbf{f}$ and compatibility $\mathbf{0} = \mathrm{inc}(\boldsymbol{\varepsilon}(\boldsymbol{\sigma}))$
- $\boldsymbol{\sigma} \in H^1$: symmetric tensor-valued

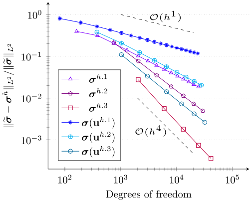
$$
\begin{aligned}
0 = & \frac{1+\nu}{E} \int_{\Omega} \mathrm{grad}(\delta\boldsymbol{\sigma}) : \mathrm{grad}(\boldsymbol{\sigma}) - \delta\boldsymbol{\sigma} : 2\mathrm{sym}(\mathrm{grad}(\mathbf{f}))\,\mathrm{d}x \\
& + \frac{1}{E} \int_{\Omega} \mathrm{div}(\delta\boldsymbol{\sigma}) : \mathrm{grad}(\mathrm{tr}(\boldsymbol{\sigma})) + \mathrm{grad}(\mathrm{tr}(\delta\boldsymbol{\sigma})) : \mathrm{div}(\boldsymbol{\sigma})\,\mathrm{d}x \\
& - \frac{1+\nu^2}{E(1-\nu)} \int_{\Omega} \mathrm{tr}(\delta\boldsymbol{\sigma})\,\mathrm{div}(\mathbf{f})\,\mathrm{d}x - \frac{1}{E} \int_{\partial\Omega} \delta\boldsymbol{\sigma} : \bar{\boldsymbol{\tau}}\,\mathrm{d}s + \text{stab}
\end{aligned}
$$

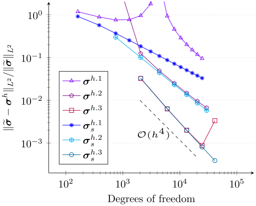
Challenges

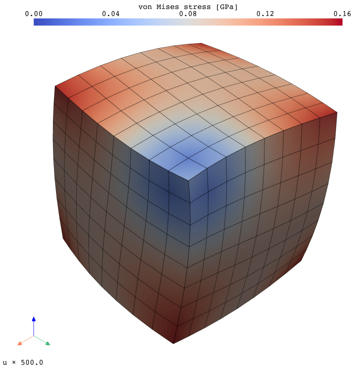
- nothing, really.

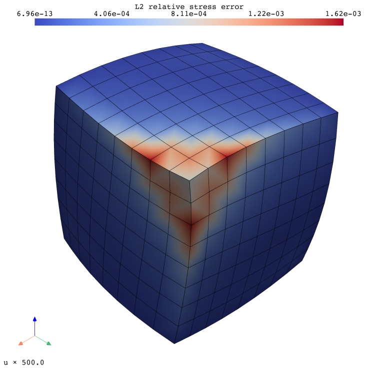
Stress-only form | demo

- cube domain
- manufactured solution
- D-only and D/N boundary conditions tested









Formulation and discrete function spaces

- see work by Simo [3] and others
- express state in principal space → Ogden-type material, $W(\boldsymbol{\lambda})$
  $\mathbf{C}(\mathbf{u}) = \sum_k^3 c_k \, \mathbf{E}_k$ with $\mathbf{c} = [c_1, c_2, c_3] = \boldsymbol{\lambda}^2$
- displacement-based, $\mathbf{u} \in H^1$

Spectral form | hyperelasticity, isotropic

Formulation and discrete function spaces

- see work by Simo [3] and others
- express state in principal space → Ogden-type material, $W(\boldsymbol{\lambda})$
  $\mathbf{C}(\mathbf{u}) = \sum_k^3 c_k \, \mathbf{E}_k$ with $\mathbf{c} = [c_1, c_2, c_3] = \boldsymbol{\lambda}^2$
- displacement-based, $\mathbf{u} \in H^1$

$$0 = -\int_\Omega \delta\boldsymbol{\lambda}(\mathbf{u}) \cdot \mathbf{p}(\boldsymbol{\lambda}(\mathbf{u}))\, \mathrm{d}x + \int_\Omega \delta\mathbf{u} \cdot \mathbf{f}\, \mathrm{d}x + \int_{\partial\Omega_t} \delta\mathbf{u} \cdot \bar{\mathbf{t}}\, \mathrm{d}s \quad \text{with} \quad \mathbf{p}(\boldsymbol{\lambda}) = \frac{\partial W(\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}}$$

Spectral form | hyperelasticity, isotropic

Formulation and discrete function spaces

- see work by Simo [3] and others
- express state in principal space → Ogden-type material, $W(\boldsymbol{\lambda})$
  $\mathbf{C}(\mathbf{u}) = \sum_k^3 c_k \, \mathbf{E}_k$ with $\mathbf{c} = [c_1, c_2, c_3] = \boldsymbol{\lambda}^2$
- displacement-based, $\mathbf{u} \in H^1$

$$0 = -\int_\Omega \delta\boldsymbol{\lambda}(\mathbf{u}) \cdot \mathbf{p}(\boldsymbol{\lambda}(\mathbf{u}))\,\mathrm{d}x + \int_\Omega \delta\mathbf{u} \cdot \mathbf{f}\,\mathrm{d}x + \int_{\partial\Omega_t} \delta\mathbf{u} \cdot \bar{\mathbf{t}}\,\mathrm{d}s \quad \text{with} \quad \mathbf{p}(\boldsymbol{\lambda}) = \frac{\partial W(\boldsymbol{\lambda})}{\partial\boldsymbol{\lambda}}$$

Challenges

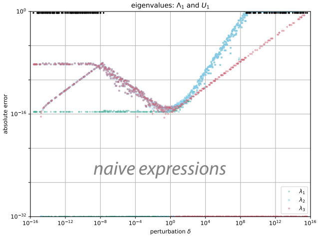
- (symbolic) spectral decomposition of 3x3 matrices in UFL
- automatic differentiation through eigenvalues (and eigenbases)
- reduced numerical accuracy in case of coalescing eigenvalues

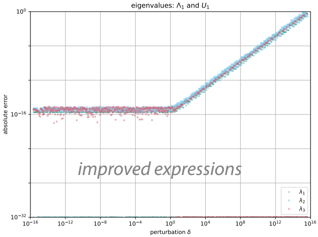
Spectral form | solved challenges

Symbolic spectral decomposition of 3x3 matrices (over reals)

- based on our work in [4]
- expressions for general matrices
- reduce rounding errors & catastrophic cancellation in matrix invariants
- eigenvalues: c_k always ordered
- eigenprojectors: by differentiation $\mathbf{E}_k = \dfrac{\partial c_k}{\partial \mathbf{C}^{\mathsf{T}}}$
- implementation in UFL

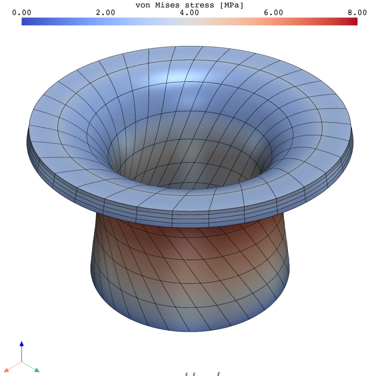
```python
import ufl
from dolfiny.invariants import eigenstate

F = ufl.Identity(3) + ufl.grad(u)
C = F.T * F

# eigenvalues and eigenprojectors of C
c, E = eigenstate(C)

# principal stretches
λ = ufl.as_vector(ufl.elem_op(ufl.sqrt, c))
```



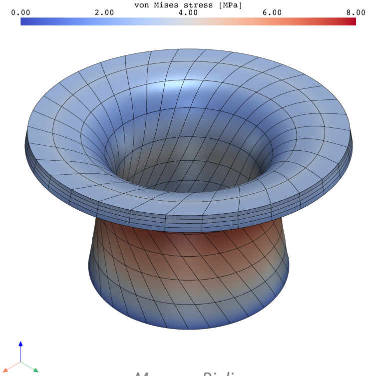


Spectral form | demo

- tube, upright
- fixed bottom face
- torque applied to upper face



neo-Hooke



Mooney-Rivlin

Summary and outlook

`https://github.com/fenics-dolfiny/dolfiny`

`pip install dolfiny`

Discussed topics available as demos

- `demo/tdnns`
- `demo/beltrami`
- `demo/spectral`

Related functionality provided with `dolfiny`

- custom Basix element: HHJ/PS on triangles/tetrahedra
- projector-restricted expressions over codimension-1 domains
- (sophisticated) spectral decomposition using UFL

References

[1] Astrid Pechstein and Joachim Schöberl. 'Tangential-displacement and normal-normal-stress continuous mixed finite elements for elasticity'. In: *Mathematical Models and Methods in Applied Sciences* 21.08 (2011), pp. 1761–1782. DOI: 10.1142/S0218202511005568. eprint: https://doi.org/10.1142/S0218202511005568. URL: https://doi.org/10.1142/S0218202511005568.

[2] Adam Sky and Andreas Zilian. 'Symmetric unisolvent equations for linear elasticity purely in stresses'. In: *International Journal of Solids and Structures* 295 (2024), p. 112808. ISSN: 0020-7683. DOI: https://doi.org/10.1016/j.ijsolstr.2024.112808. URL: https://www.sciencedirect.com/science/article/pii/S0020768324001677.

[3] Juan C. Simo and Robert L. Taylor. 'Quasi-incompressible finite elasticity in principal stretches. Continuum basis and numerical algorithms'. In: *Computer Methods in Applied Mechanics and Engineering* 85.3 (1991), pp. 273–310. ISSN: 0045-7825. DOI: https://doi.org/10.1016/0045-7825(91)90100-K. URL: https://www.sciencedirect.com/science/article/pii/004578259190100K.

[4] Michal Habera and Andreas Zilian. *Symbolic spectral decomposition of 3x3 matrices*. 2021. arXiv: 2111.02117 [math.NA].